WITN05970400

Witness Name:	Jeremy Peter Folkes
Statement No.:	WITN05970400
Dated:	28th May 2024

POST OFFICE HORIZON IT INQUIRY

FOURTH WITNESS STATEMENT OF JEREMY PETER FOLKES

I, Jeremy Peter Folkes, will say as follows:

INTRODUCTION

1. This is my fourth statement to the Post Office Horizon IT Inquiry ("the **Inquiry**"), in which I provide further information on Riposte that I consider may be relevant to the Inquiry. I contacted the Inquiry myself to offer this information after following hearing oral evidence given during Phase 3. The Inquiry asked me to formalise any additional information I wished to provide in the form of this, my fourth witness statement[1].

2. I assembled the Technical Information here having been following, in some detail, the Phase 3 Hearings and Witness Statements and feeling that in some areas

[1] This statement was substantively complete in summer 2023 however it has taken longer than expected to be reviewed by the Inquiry and returned for signature, hence this signed version is only being submitted in May 2024. However, it still primarily contains information intended to be of relevance to Phase 3.

WITN05970400

such additional information might be useful to the Inquiry. I believe that the generic knowledge I had might bring some clarification to matters under investigation.

3. I described my Professional Background up to the point of leaving the Post Office in 2000 in some detail in my first statement [WITN05970100]. As I mentioned in paragraph 6 of that statement after a short period in another company (**Anshe Ltd**), following a takeover I ended up working for **Escher Group Ltd** where I remained for some 21 years until my retirement in 2021. Over that period, I gained some considerable technical and operational knowledge of the underlying Escher/Riposte product across other postal automation projects around the world (primarily outside of the UK), and believe that some of this knowledge may be useful to the Inquiry.

4. The level of detail given here would be generally known to the senior technical staff at Escher's larger customers and reflects information Escher would have provided to its customers through technical training. My intention is not to stray into any 'trade secrets' of Escher.

5. For the avoidance of doubt I have no ongoing contact or relationship with Escher (although I left on good terms and still have occasional social contact with a few ex-colleagues). Specifically, I do not have any visibility of whether Escher has been approached by the Inquiry or is contributing to the Inquiry, and am not speaking on behalf of Escher in this statement. I also do not have ongoing access to any of Escher's internal systems which may hold information on specific bugs or product releases over the years.

WITN05970400

6. Within this statement, unless I explicitly specify otherwise, I will be describing the generic Riposte product (of which I can legitimately claim detailed knowledge, as explained in the next paragraph) rather than the specific Pathway/Fujitsu implementation for Horizon (of which I have had little visibility since the leaving the Post Office in 2000 and did not/do not have authoritative knowledge).

7. I have been asked by the Inquiry to explain why I can legitimately claim detailed knowledge of Riposte. I worked with Riposte for over 20 years, becoming the *de facto* lead in the company (outside of the core US team) in live operation of the Riposte Message Server, including undertaking many consultancy assignments with customers and being responsible for creating and delivering technical training on the Riposte product's System Administration for many years. During that time, I worked with over 15 postal authorities worldwide in detail on their systems and a number of others on a more occasional basis or for pre-sales. Latterly, I held the post of Chief Technology Officer (CTO) and then Chief Product Officer (CPO) in Escher, before finally working as Principal Consultant.

8. Finally, in this formal Witness Statement I have added some additional narrative on my memories of the operation of the support arrangements between Escher and Fujitsu (although I do not believe I ever had responsibility for such arrangements with Fujitsu on Horizon).

WITN05970400

RIPOSTE MESSAGE SERVER – BACKGROUND INFORMATION

9. Within a Riposte-based system (such as Legacy Horizon, or any of the other 20-30 Posts who use/have used it), the underlying software component provided by Escher is the **Riposte Message Server** (frequently just known as **Riposte)**. For some periods in the 2000s the Message Server was also known as**WebRiposte**; initially WebRiposte was sold as an upgrade to Riposte (with some additional web-related facilities) but after a while the upgraded product was reverted to use the simpler Riposte name. For the purpose of this discussion the two can be treated as synonymous.

10. Escher also provided[2] other components, which were used in various combinations by different customers, in the "application" space. These included the **Riposte Desktop** (providing the User Interface (UI), peripheral device support and other such functions in Horizon), certain business applications (including **Mails**, a pricing assessment feature for mailing, which was introduced in Horizon around 2003). There were also several other items which were not used by Fujitsu in Horizon. All these components sat "above" of the Message Server but the Message Server is/was not dependent on any of them. Note that the packaging/bundling and naming of Escher components changed somewhat over the years, mainly for marketing/commercial reasons, but this does not change the general concept.

[2] I use the past tense here as Escher has since refreshed the application part of their offering and the Riposte Desktop is, I believe, no longer actively sold (although at the point I left in 2021 it was still supported and used by some customers.

WITN05970400

11. **Riposte** can be described is a **message-orientated middleware**[3], a software layer designed to provide data storage and transmission facilities to local applications in a large distributed estate, in manner which maximises the availability of data on the local terminal and allows the terminal to operate autonomously even if other terminals in the office, the network or the central systems are temporarily unavailable. Riposte handles the replication of data between terminals and to/from the data centres, on a "**best-efforts basis**[4]", whilst allowing service to customers to continue even in times of failure, and with data backed up across multiple terminals whenever possible, in a matter which is largely transparent to the local application. Data in this context includes both transaction data (for transactions performed in an office) and reference data (such as prices, product names etc).

12. This component runs as a **Windows NT Service** (that is, a program which is started automatically by Windows when the machine boots and effectively runs in background), and has no visible User Interface. This Riposte service runs whether or not an application has been started or whether anyone is logged on to the terminal, and as a service runs as a separate process to the user application.

---

[3] A common description of middleware (this from Microsoft) is " *Middleware is software that lies between an operating system and the applications running on it. Essentially functioning as hidden translation layer, middleware enables communication and data management for distributed applications. It is sometimes called 'plumbing' as it connects … applications…… .*". Middleware is literally "in the middle" between the application (on top) and the operating system (underneath) and is therefore generally not visible to the application user, but is providing a set of common functions required by the application or applications; in the Riposte case providing communications and data storage.

[4] By "Best-Efforts Basis" we mean the protocol will persistently attempt to replicate data, even over poor quality and intermittent communications lines and with intermittently available endpoints, including retrying multiple times, attempting multiple routes to a destination, and several other features, rather than "giving up" (failing) if the first attempt or few attempts fail.

WITN05970400

13. On a counter workstation, Riposte's primary task is to provide and store data on behalf of the application program, which in Horizon was based on the **Riposte Desktop** (frequently just known as the **Desktop**), within which the specific Post's business applications are implemented – in Horizon this would include EPOSS[5], APS, OBCS, etc as implemented by Pathway/Fujitsu).

14. However, the Riposte instance on each workstation acts also as part of the overall community of workstations in an office (known as a **Riposte Group**), providing a backup on each workstation of all data created by all the other workstations in the group or from the centre. In this way the set of workstations in a group effectively manage a shared data store for the group.

15. The Riposte service is responsible for:

    a. Managing data access to the local **Riposte Message Store,** held as a single file on the hard disk on counter terminal; Riposte is responsible for reading/writing data from/to the message store, at the request of the application (via an **Application Programming Interface** or **API**). Note that the application never writes to the Riposte Message Store directly, it always accesses via the local Riposte Message Server.

    b. **Housekeeping** on the Riposte Message Store, including space management, archiving of expired data, and maintenance of indexes to

[5] EPOSS Electronic Point of Sale Service, APS Automated Payment Service, OBCS Order Book Control Service – all business applications services in Horizon.

WITN05970400

aid performance for retrieval. Data is retained for a defined **expiry period**, which I believe on Horizon was initially set to 42 days (although the out-of-the-box default for Riposte is 90 days).

c. **Replicating and synchronising data** with neighbouring Riposte Message Servers (usually on all other counter workstations in an office) over the Local Area Network (LAN), or to a **Mirror** Riposte Message Server (on a second disk in the same machine, in a single terminal office).

d. **Replicating and synchronising data** with one or more central Correspondence Servers over the Wide Area Network (WAN), if configured. In Horizon I believe just one counter terminal in each office was configured in this way, but other configurations are used elsewhere.

e. **User Management**, providing underlying facilities for user and user group management, including password validation and facilities for access control.

16. The same Riposte Message Server software runs on the central **Correspondence Servers** (CS), which consolidate data from many offices (up to the whole estate); for resilience purposes there are typically 4 CS spread over at least two sites. In Legacy Horizon I believe (at least initially) the estate was divided into 4 clusters of CSs, each of 4 servers, spread over two sites. Correspondence Servers typically have an array of large disks (typically up to

WITN05970400

16) as they hold data for many offices, and Riposte manages the allocation of office data across these disks.

17. On the Correspondence Servers, data is accessed by software programs called **Agents** which act as the link between Riposte and third-party systems. These use Riposte APIs into Riposte to harvest data replicated up from offices for export, and to insert data from external systems for replication down to offices. These APIs also support on-line/real-time operation if needed, by listening for incoming messages for on-line processing.

18. Data in Riposte is stored in what are called **Messages**, based a self-describing format known as **Attribute Grammar**, made up of **Name/Value** pairs (written as `<Name:Value>`, for instance `<Price:4.56>`)

19. All messages have some standard content, defined by Riposte and not modifiable by the application, including the following attributes:

    a. **GroupId** – office number, unique across the Post's estate.

    b. **Id -** node id – the terminal number in an office (1-31), or server number (32-63) for CS-written messages[6]. Other discrete ranges exist in later product versions but are not relevant here.

[6] Note that any messages created and inserted at the CS will have the CS's Id (node) number and so should be identifiable as not having been created by the office. This of course would not apply if (hypothetically) remote

WITN05970400

c. **Num -** a unique message number for that group/node pair, which is allocated in a strictly monotonically increasing manner. On this basis, every message across an entire estate has a unique identifier made up of GroupId, Id and Num.

d. **Date/Time** of creation of the message – in UTC (effectively GMT).

e. **UserId** - if a user was logged on to Riposte at the time of creation.

f. **Cyclic Redundancy Check (CRC)** – effectively a strong checksum - computed on the entire contents of the message at the time of creation. This CRC is recomputed and checked by Riposte whenever a message is retrieved or received by the Message Server to detect corruption of a message in storage or transit.[7]

20. Applications can then specify their own payload data to be held within each message (as long as it does not conflict with one of the above reserved terms); although no formal data schema is enforced by Riposte, it is good practice for application designers to "design" the set of message types which they are going to use and for what purposes, and typically to add a data field called Application

access via a third-party product is made to the terminal, where a message is created and inserted on that terminal via such access.

[7] Note that CRCs are also used extensively within the internals of the Riposte Message Store to detect any corruption of the contents on disk, in addition to being used to protect individual messages.

WITN05970400

to show which application has created the data (e.g., `<Application:EPOSS>` for an EPOSS message)[8].

21. Riposte also supports the concept of **Objects** (implemented on top of the Message concept described in the previous paragraph) for static data or data which has an indefinite retention period. These have the same standard attributes as outlined in the previous paragraph but also include a versioning mechanism which can be used to allow data values (such as reference data) to be updated (for instance, for price changes). Objects can be used to implement the equivalent of a database table structure.

22. Core to Riposte is the concept of replication between neighbours, which is the means by which all the Message Servers in a group remain in synchronisation. This operates through two mechanisms, firstly **forward broadcast** (where by each Message Server in a group sends each new message onto its **neighbours**), and secondly **marker exchange** where periodically (eg every few seconds) Message Servers exchange status information with their neighbours and neighbours request any missing messages. Any messages which might have been either lost or corrupted in transit over the network are automatically **requested** when either the next message in sequence is received or the next marker is received. This includes messages which are **discarded** by Riposte due to some incorrect contents. This mechanism is effectively self-healing and handles synchronisation after network errors, power outages or

[8] Typically applications will introduce intermediate software layers with APIs to handle specific writing of specific message types, as a means of enforcing consistency, however I cannot comment on whether this was done in the Horizon.

WITN05970400

other terminal downtime, and is also used for recovering missing messages after a disk replacement.

23. Messages are considered **immutable**, that is once written they cannot be changed – so that a message with a particular GroupId/Id/Num identity has the same contents, including the same Date/Time and CRC, wherever it has been replicated. If data needs to be updated, then a new message must be written by the application (with a new identity), and so typically data which needs to be updated will use Objects (with versioning) as discussed earlier.

LOGGING IN RIPOSTE SYSTEMS

24. There has been various discussion in Phase 3 regarding logging mechanisms in Horizon, and I felt in would be useful to outline the options usually available in a Riposte system.

Use of Riposte for Logging

25. Within a Riposte system, the primary data store available for applications is the Riposte Message Store, which benefits from replication to neighbouring terminals and to/from the Correspondence Servers. This is where a business application (such as Horizon's EPOSS) is expected to write all transaction or accounting information, and provides the underlying "audit trail", with benefits of sequential numbering, date/time stamping, replication and immutability.

WITN05970400

26. The data is automatically replicated to the central Correspondence Servers, as mentioned earlier. In Horizon, I believe a custom software **agent** was responsible for taking data daily from the Correspondence Servers and writing this to longer term storage outside of Riposte (and which I believe from where ARQ Requests were taken).

27. The Riposte product itself write a small number of **events** itself to the Riposte Message Store, including an event at startup and events for logon/logoff and access control or configuration changes. However the majority of information written by Riposte is being initiated by and stored at the request of the application (eg Horizon's EPOSS). This is equivalent to an application writing to a database. Riposte knows nothing of the meaning of this data, it is just storing it on behalf of the application.

Window NT Event Logs

28. **Microsoft Windows NT**, the underlying operating system used by Horizon, provides a facility known as the **Event Log**, which is used by both Windows itself and by applications to record events in a common format. Events are stored in local files managed by Windows, and include a date/time stamp, severity, source, event number and other details.

29. Severity, assigned by the application generating the event, can be one of **Information (Blue), Warning (Yellow)** or **Error (Red),** allowing applications to categorise the information they produce for ease of analysis.

WITN05970400

30. There are two main Event Logs used in Windows NT – **System** and **Application** (in addition to a **Security** event log which is of less interest here):

    a. **System** – this is used purely by core Windows itself to record events related to Windows, including features like services starting and stopping. Of interest on a Riposte system would be events for Windows stopping and starting the underlying Riposte Service (and such events are initiated by Windows).

    b. **Application** – this is used by applications (in this case this includes Riposte Message Server, which Windows considers to be an application) to record events of their own definition. The Riposte Message Server makes considerable use of the Windows Application Event log to record events from each of its subsystems, across the three available categories of Information, Warning and Error.

31. Event Logs are by default only stored on the local machine, however third-party products are available which will monitor and harvest event log information for system management purposes. I believe that Tivoli had been proposed by Pathway for this purpose on Horizon, but I have no information on how this was used.

32. Event Logs are normally configured with a Maximum log size (to avoid filling the disk) and generally set to “Overwrite events as needed (oldest events first)” but I do not know what settings were used on Horizon.

WITN05970400

Other Logs

33. The Riposte Desktop had a component known as the Peripheral Server, which was responsible for handling input and output with counter devices (eg barcode scanners, card readers, printers etc), and this could be configured to write logging information to text files on the local hard disk. I believe these files may have had names "**pslog**" or "**psstandard**" or similar, and were intended more for application debugging during development rather than live use.

34. There was also mention by one witness in Phase 3 of a specific "audit log" file on the counter terminal. From searching the transcripts I believe this was Anne Chambers on 2nd May 2023. I do not recognise this as an Escher facility and therefore it seems likely this was something added by Fujitsu at application level.

35. Several witnesses have been questioned in Phase 3 regarding "key-stroke logging" (including I believe Mike Peach, Gary Blackburn and Anne Chambers). I am not aware of any key-stroke logging implemented in Riposte or the Escher Riposte Desktop in the lifetime of Legacy Horizon[9]. I do believe a facility was discussed and potentially added by Escher to their product in the early 2010s for use with a major customer (not related to the UK Post Office) for a Self-Service Kiosk project, where it was required to record exactly what the end-user had typed.

[9] Note that key-stroke logging has to be used with care to avoid compromising security by capturing sensitive information such as passwords or account numbers which could be mis-used. Full keystroke logging should not be introduced without due consideration of such security risks and implementation of suitable protection. In this context I believe Escher viewed it as more applicable in a Self-Service environment to help manage customer disputes, rather than on a Counter Terminal.

WITN05970400

This suggests that this was not present in the versions used on Horizon in the 2000s.

36. I have been asked by the Inquiry for my view on whether key-stroke logging existing in Horizon in the early 2010s. I cannot comment on this as by 2010 Legacy Horizon had been replaced by Horizon Online, which I believe did not use any of the Escher Riposte product set, and in which I had no involvement.

RIPOSTE EVENTS

37. As mentioned above, the Riposte Message Server makes considerable use of the Windows NT Application Event Log to report events, both **expected** (eg starting up, or routine overnight activities taking place, or low disk space) and **unexpected** (where some internal process has failed unexpectedly or hit some other problem). In the latter case the event information may need to be reported to Escher for interpretation and advice, and the events form an important part of diagnosis.

38. There have been two cases in recent Phase 3 testimony where reference has been made to specific Riposte (Application) event log messages.

Lock Events

39. Mention has been made during the Hearing on 3rd May 2023 (Anne Chambers) and elsewhere of various **Lock Events** raised by Riposte, discussed in the context

WITN05970400

of the so-called Callendar Square bug. I felt it may be helpful to provide more information on the concept of locking.

40. Riposte (the Riposte Message Server) is a complex software component which uses the software engineering concept of **multi-threading** to able to perform multiple disparate tasks in parallel. As outlined earlier, Riposte may be handling requests from the application, replication to/from various neighbours, indexing and housekeeping tasks, etc, all at the same time, and will allocate different **threads** for different purposes. There even be multiple threads for the same category of tasks (e.g. multiple threads for retrieving messages during message replication), to allow operations to take place in parallel and to optimize use of the computer's hardware resources. This latter case is most frequently used when Riposte is used on Correspondence Servers (centrally) on hardware with multiple disks, but can also be relevant on counter terminals and is included here for completeness

41. However, using multi-threading means that care needs to be taken to manage **concurrent access** to the same resource – for instance, to avoid a conflict if multiple threads need to update the same block on disk. This is achieved by using **locking**, where by one thread can request a lock for a specific resource so that it has sole rights to use that resource for the duration of an operation. It will **take the lock**, make its update, and then unlock the resource (**drop the lock**). Other threads which happen to be wanting to take a lock on the same resource when one has it locked would be **waiting for lock**, and typically they would wait for a period and then get a **timeout waiting for lock.**

WITN05970400

42. Locking is used extensively within Riposte to manage concurrency across a range of different data structures and functions, and therefore a range of different (and unrelated) modules may (theoretically) raise a timeout waiting for lock error.

43. Timeouts waiting for locks can occur for a number of reasons, and are rightly reported by Riposte in the Event Log so that they can be investigated, along with whatever activity failed. If repeated such events occur, is it likely that some key function is failing to succeed (or at least to succeed in a timely fashion).

44. Possible reasons for a timeout waiting for lock would be if a thread which takes a lock fails (crashes) for some reason, leaving a lock in place – which would be cleared on a restart – or an overloaded/under-resourced machine where many threads are competing for the same resource. In the "Callendar Square" bug it appeared that a repeated timeout waiting for lock due to some issue prevented replication in a specific direction, until a restart.

45. Care should therefore be taken not to associate all events related to "locks" as indicative of a common fault or problem; locking is a common mechanism used widely across Riposte, and the presence of various events involving locks is not unexpected. As an analogy, a car may have many fuses, and not all cases of a fuse blowing would be down to the same underlying problem.

46. However, from what I can see Calendar Square bug (KEL B259) did appear to be related to a genuine issue and did delay replication until a reboot, which clearly did have serious repercussions on successful balancing.

WITN05970400

Discards

47. During the 10th May 2023 Hearing (Stephen Parker), a document entitled "HORIZON KEL B259" was discussed. This talks of an event raised by Riposte which included the text "**Invalid Node identifier**" and referred to a message being "**Discarded**". The discussion with the witness may have led to the conclusion that data would be lost by this, however from my product knowledge I believe this is unlikely to be the case.

48. This is a good example of Riposte using the Event Log to report a condition would was unexpected and certainly (as with any Error event) should have been investigated by Fujitsu and/or discussed with Escher. However, a one-off occurrence would not necessarily have affected live operation, in that Riposte's replication protocol would have detected a missing message and requested a replacement copy – repeatedly if necessary – from its neighbours. If an **event storm**[10] of such events had been recorded this would have indicated a more serious, operation-affecting, issue, however a single occurrence would have only momentarily delayed the message being replicated

49. The presence of this event potentially indicates a bug which, if occurring widely, could affect the efficiency of replication (and could force repeated re-requests,

[10] The term "**event storm**" is used to describe the situation when a large number of events – a storm – occurs in a very short timescale. The presence of an event storm would generally suggest that a serious repeated failure has occurred and was not being handled gracefully by the software.

WITN05970400

increasing network traffic), however at low incidence it is likely that was handled silently by the protocol.

Escher Support for Pathway/Fujitsu

50. There has been various mention in Phase 3's Hearings by former Fujitsu employees of the technical support arrangements between Escher and Pathway/Fujitsu, for the use of the Riposte product. Although support for Fujitsu was not part of my role, I do have some memories of how it worked which may be of use to the Inquiry.

51. I do not recall the exact commercial arrangements between Escher and Fujitsu (and may have never had detailed knowledge) for support, but it was normal practice to have a **Maintenance Agreement** (which provided fault fixing and regular software updates) and a **Support Agreement** (which provided additional help and support for both development and operations). I remember for some periods in the 2000s there was also a **Consultancy Agreement** to enable Fujitsu to call off assistance on more significant tasks if needed, although I do not remember this being used by them to any significant extent.

52. My recollection was that Fujitsu, as Escher's largest and highest profile customer at the time, initially had priority contact between their technical contact (who I believe was **Mark Jarosz** initially) and the Message Server team at Escher's then HQ in Cambridge, Massachusetts, bypassing the Customer Support routes used

WITN05970400

by other customers (who were mostly fronted via either the Dublin or UK Escher offices).

53. I believe that, at least initially, Mark Jarosz had direct access, if needed, to **Andrew Sutherland (Drew Sutherland)**, who was co-founder of Escher and the original designer and architect of Riposte. I believe this "Mark Jarosz - Drew Sutherland" link was mentioned in various exhibits shown in Phase 3. Drew moved on from Escher (into academia I believe) at some point, possibly around 2003, but I cannot remember whether Mark Jarosz had continued access to others in the Cambridge team at that time.

54. This "Gold" access reflected the importance to Escher of the UK Post Office/Horizon project, and the considerable level of expertise/analysis offered by Mark Jarosz on the product (allowing the lower levels of support "filtering" to be circumvented). By this I understood that if Mark Jarosz reported an issue, it would be considered to have been already investigated in serious detail on the customer side (which did not always happen with every customer), and it would get urgent attention in Escher.

55. At some point Fujitsu were migrated to use the same support route into Escher as other customers, fronted by dedicated Escher Customer Support staff in Cambridge and Dublin or the UK. I cannot remember exactly when this happened, but I do believe, however, that Fujitsu's support was fronted out of the UK by 2005. This would not have meant however that Fujitsu did not still get an expedited service, given their importance.

WITN05970400

56. Customers used an online portal ("**Escher On-Line**" or **EOL**) for issue reporting and tracking, and this was used both for live and development issues. The version of EOL which I knew and which Fujitsu would have used was introduced in Escher in 2001 (based on a commercial product known as TeamTrack) and would have been in place until well after Legacy Horizon was replaced. An archive of this system was still available in Escher at the time I retired in 2021.

57. I cannot remember (if I ever knew) what system was in place before mid-2001 (prior to EOL) which might have been used by Pathway[11]; as indicated earlier there were special arrangements in place for Pathway which allowed direct access into the US team. I presume some system existed from 1995 onwards all through all the development years for Horizon, and this may have persisted to 2001.

58. I remember that the demands of Fujitsu/Horizon would drive the priority for bug fixes and releases for new versions of software, and that Escher would provide releases targeted for Fujitsu to fix a specific issue if considered urgent, again reflecting the importance of the UK Horizon system to Escher. Escher also maintained a special "UK only" branch of the software for Horizon at one time, containing only changes required by or agreed to by Fujitsu.

59. However, I remember though Fujitsu's lead time for getting updates from Escher into live operation in POL was long (far longer than we experienced with other customers). This was, I believe, partly due to competition for both testing and

[11] In reviewing this I have a very vague recollection that a Lotus Notes system may have existed, but was not used company-wide.

WITN05970400

release “slots” on Horizon, with the number of new business features being deployed throughout the 2000s, and partly a risk-averse approach given the estate size. In particular, deploying new software to 40,000 terminals across 20,000 sites over the network used in the early 2000s would not have been trivial.

60. I believe this did mean that at times Fujitsu would report or follow-up on issues which had already been fixed by Escher, and where an update to the core software had already been made available to customers. Whilst this is a risk in any software support arrangement it can be exacerbated by the extended deployment lead time.

61. I do remember that the number of support and maintenance calls from Fujitsu had reduced to very low numbers by around 2006 and I do not believe any new issues were raised with Escher though EOL by Fujitsu after early 2007. This was of course the time when Fujitsu had started development work on the replacement Horizon Online, which did not rely on the Riposte product set (although this was apparently not fully deployed until 2010).

Statement of Truth

I believe the content of this statement to be true

Signed GRO

Dated 28th May 2024

WITN05970400

Index to Fourth Witness Statement of Jeremy Peter Folkes

No	URN	Document Description	Control Number
01	WITN05970100	Jeremy Folkes – First Witness Statement	WITN05970100